

**RESOLUCION**

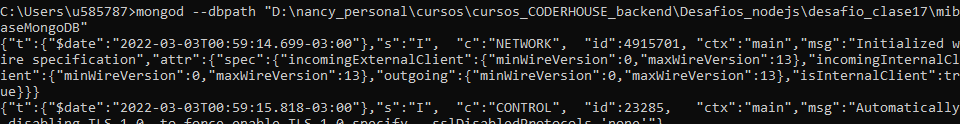
**PASOS PREVIOS**

* **SERVIDOR : inicializo base de datos de MongoBD en una carpeta creada por mi**

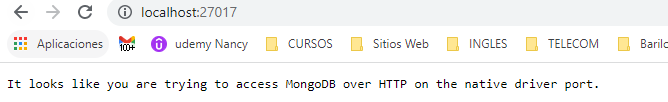
1. Creo una carpeta donde quiero q este mi base de datos , ejemplo

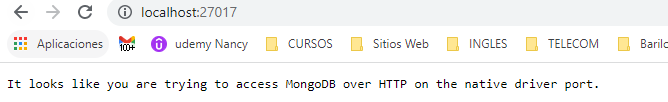
“D:\nancy\_personal\cursos\cursos\_CODERHOUSE\_backend\Desafios\_nodejs\desafio\_clase17\mibaseMongoDB”

1. Ingreso a la terminal (cmd)
2. Ejecuto el siguiente comando 🡪 mongod --dbpath “D:\nancy\_personal\cursos\cursos\_CODERHOUSE\_backend\Desafios\_nodejs\desafio\_clase17\mibaseMongoDB”



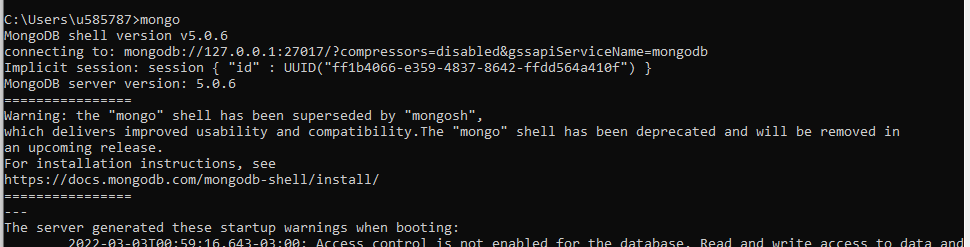
1. Se observa que en la carpeta creada ahora aparecen varios archivos.
2. Verifico que el motor de base de datos se inicie (este paso no es obligatorio, de hecho nose si esta OK jeje)





* **Cliente CLI : me conecto al servidor anterior**

1. Ingreso a otra terminal (cmd)
2. Ejecuto 🡪 mongo



Nota: Se puede usar la Aplicación “MongoDB Compass” (instalada previamente en nuestra computadora), pero el desafío pide los comandos, por lo cual lo hago directo en la terminal para probarlos.

1. crear una base de datos llamada *ecommerce* que contenga dos colecciones: mensajes y productos.

use *ecommerce*



db.createCollection("mensajes")



db.createCollection("productos")



Validación:

db



show collections



1. Agregar 10 documentos con valores distintos a las colecciones mensajes y productos. El formato de los documentos debe estar en correspondencia con el que venimos utilizando en el entregable con base de datos MariaDB.

5 Productos

db.productos.insert({codigo: "A001", fechaHora: "02/03/2022 11:01:22", nombre: "arnes", descripcion: "arnes de mujer", precio: 120, imagenURL: "/image/arnes.jpg", stock: 3})



db.productos.insert({codigo: "P001", fechaHora: "02/03/2022 11:01:22", nombre: "pedulas", descripcion: "pedulas de mujer", precio: 400, imagenURL: "/image/pedulas.jpg", stock: 5})



db.productos.insert({codigo: "A002", fechaHora: "02/03/2022 11:01:22", nombre: "arnes", descripcion: "arnes de hombre", precio: 1250, imagenURL: "/image/arnes2.jpg", stock: 4})



db.productos.insert({codigo: "C001", fechaHora: "02/03/2022 11:01:22", nombre: "casco", descripcion: "casco universal", precio: 1700, imagenURL: "/image/casco.jpg", stock: 1})



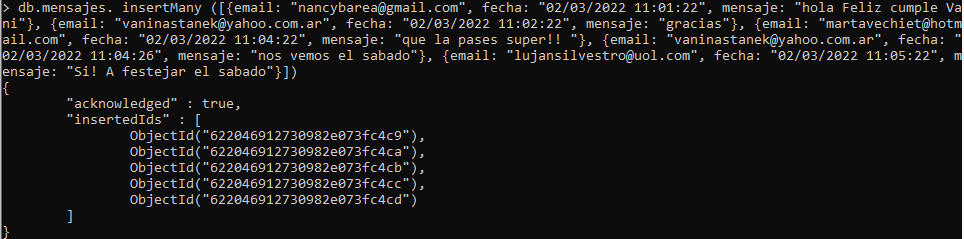
db.productos.insert({codigo: "A003", fechaHora: "02/03/2022 11:01:22", nombre: "arnes",

descripcion: "arnes de niños", precio: 3100, imagenURL: "/image/arnes3.jpg", stock: 2})



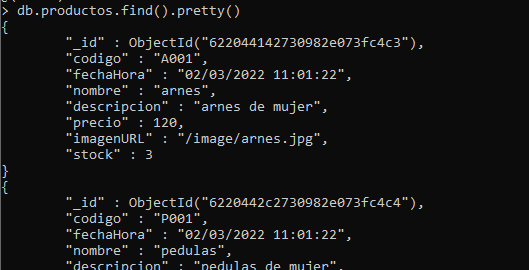
5 Mensajes

db.mensajes. insertMany ([{email: "nancybarea@gmail.com", fecha: "02/03/2022 11:01:22", mensaje: "hola Feliz cumple Vani"}, {email: "vaninastanek@yahoo.com.ar", fecha: "02/03/2022 11:02:22", mensaje: "gracias"}, {email: "martavechiet@hotmail.com", fecha: "02/03/2022 11:04:22", mensaje: "que la pases super!! "}, {email: "vaninastanek@yahoo.com.ar", fecha: "02/03/2022 11:04:26", mensaje: "nos vemos el sabado"}, {email: "lujansilvestro@uol.com", fecha: "02/03/2022 11:05:22", mensaje: "Si! A festejar el sabado"}])

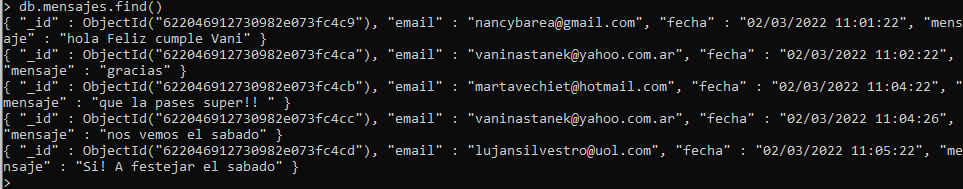


Validación:

db.productos.find().pretty()



db.mensajes.find()

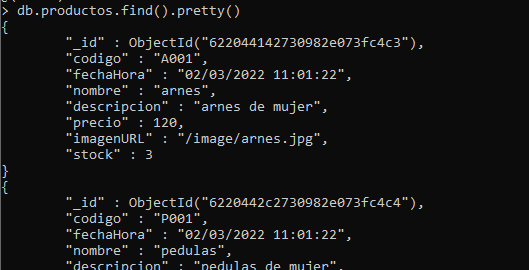


1. Definir las claves de los documentos en relación a los campos de las tablas de esa base. En el caso de los productos, poner valores al campo precio entre los 100 y 5000 pesos(eligiendo valores intermedios, ej: 120, 580, 900, 1280, 1700, 2300, 2860, 3350, 4320, 4990).

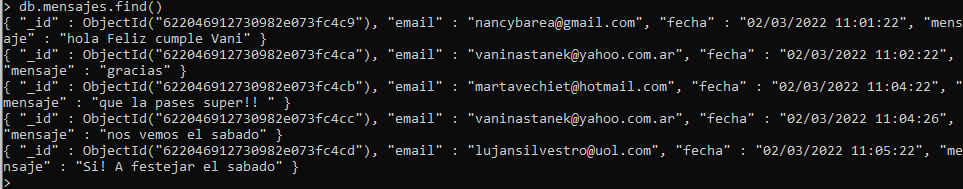
Resuelto en el punto anterior al cargar los productoss, los valores de los precios son entre 100 y 5000

1. Listar todos los documentos en cada colección.

db.productos.find().pretty()



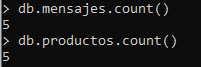
db.mensajes.find()



1. Mostrar la cantidad de documentos almacenados en cada una de ellas.

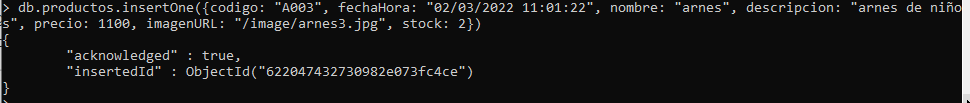
db.mensajes.count()

db.productos.count()

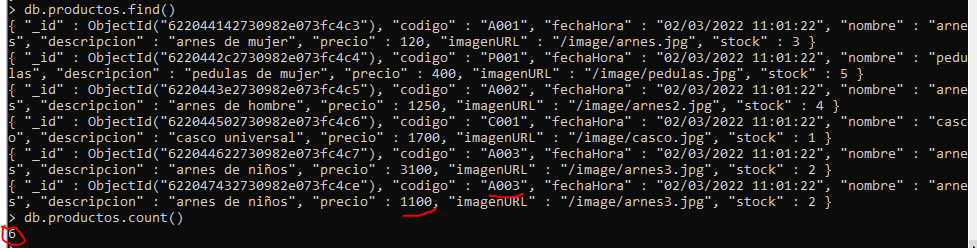


1. Realizar un CRUD sobre la colección de productos:
   1. Agregar un producto más en la colección de productos

db.productos.insertOne({codigo: "A003", fechaHora: "02/03/2022 11:01:22", nombre: "arnes", descripcion: "arnes de niños", precio: 1100, imagenURL: "/image/arnes3.jpg", stock: 2})

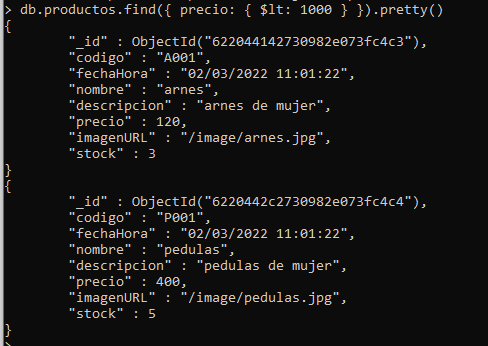


Verifico



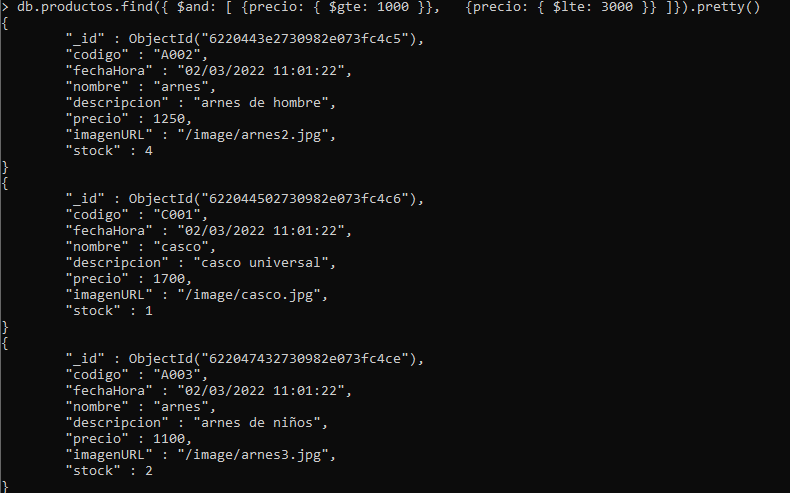
* 1. Realizar una consulta por nombre de producto específico:
* Listar los productos con precio menor a 1000 pesos.

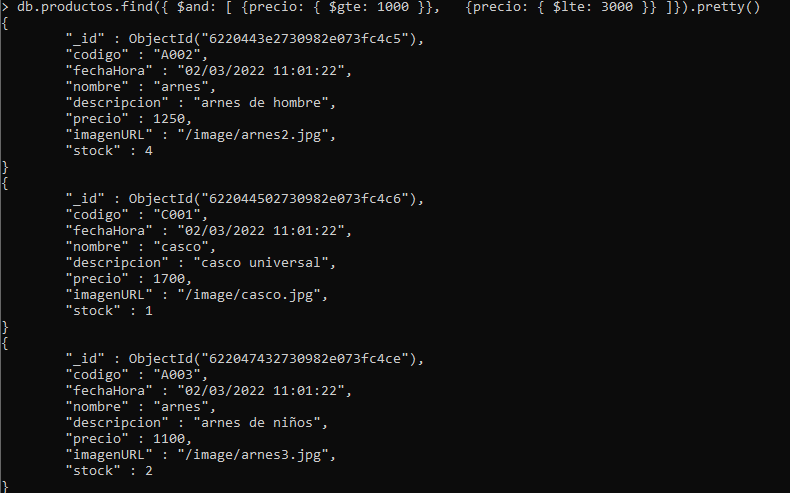
db.productos.find({ precio: { $lt: 1000 } }).pretty()

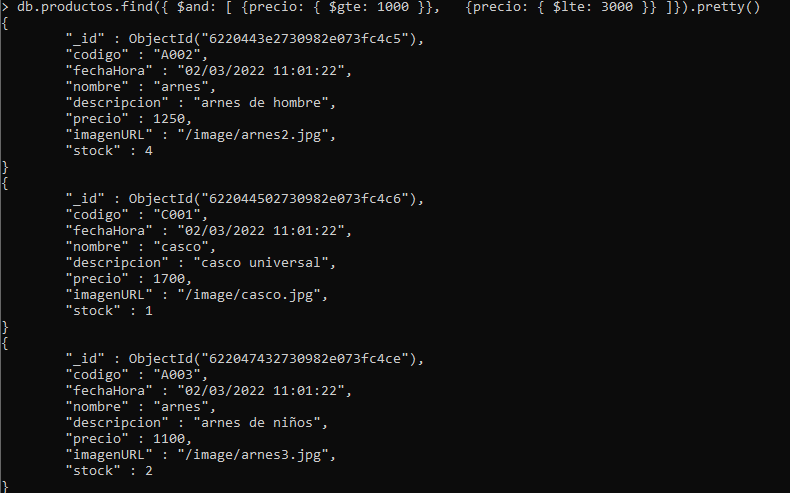


* Listar los productos con precio entre los 1000 a 3000 pesos.

db.productos.find({ $and: [ {precio: { $gte: 1000 }}, {precio: { $lte: 3000 }} ]}).pretty()

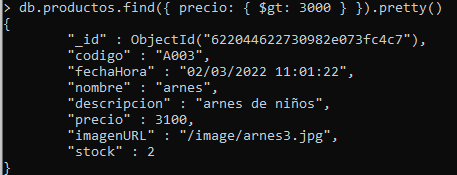






* Listar los productos con precio mayor a 3000 pesos.

db.productos.find({ precio: { $gt: 3000 } }).pretty()



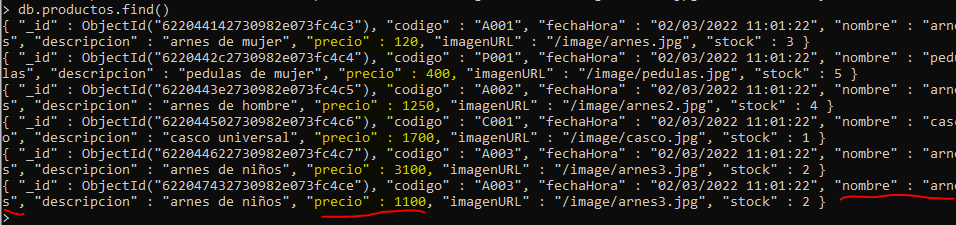
* Realizar una consulta que traiga sólo el nombre del tercer producto más barato.

db. productos.find({},{"nombre":1}).skip(2).limit(1).sort({"precio": 1})



**verifico**

listo todos



Como tengo varios con nombre “arnes” veo si me devuelve el correcto

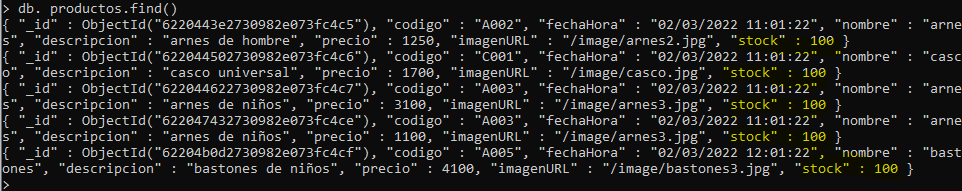


* 1. Hacer una actualización sobre todos los productos, agregando el campo stock a todos ellos con un valor de 100.

db.productos.updateMany( {}, { $set: {stock: 100}})



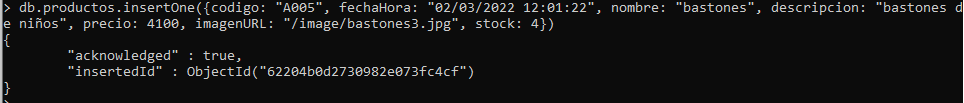
Valido



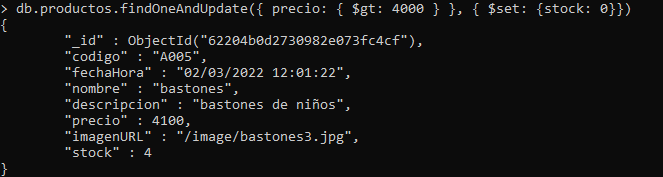
* 1. Cambiar el stock a cero de los productos con precios mayores a 4000 pesos.

como no tenía producto con mayor a 4000 pesos lo cree

db.productos.insertOne({codigo: "A005", fechaHora: "02/03/2022 12:01:22", nombre: "bastones", descripcion: "bastones de niños", precio: 4100, imagenURL: "/image/bastones3.jpg", stock: 4})

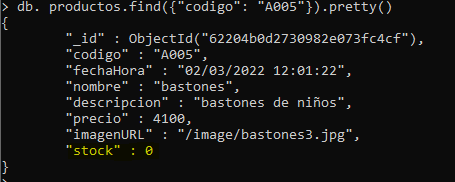


db.productos.findOneAndUpdate({ precio: { $gt: 4000 } }, { $set: {stock: 0}})



**valido**

db. productos.find({"codigo": "A005"}).pretty()

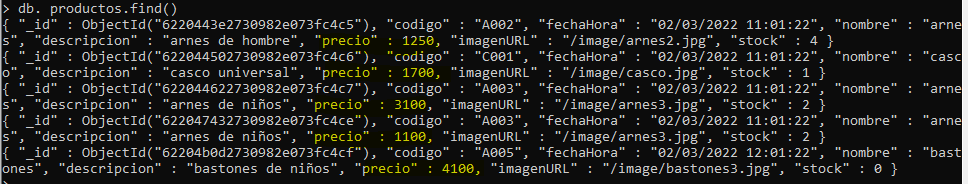


* 1. Borrar los productos con precio menor a 1000 pesos

db.productos.remove({ precio: { $lt: 1000 } })



**Valido**



1. Crear un usuario 'pepe' clave: 'asd456' que sólo pueda leer la base de datos ecommerce. Verificar que pepe no pueda cambiar la información. (OPCIONAL HACERLO PORQUE NO LO VIMOS EN CLASE)

db.createUser({ "user": "pepe", "pwd": "asd456", "roles": ["read"] })



No me funciono la validación… verlo en AFTER

